16. Method according to claim 1, wherein indicating the number of unissued words includes storing the number of issued words.

In the drawings:

Please accept the amended Figures 3-5, in which changes to the drawing sheets are shown in red.

REMARKS

The title has been amended as requested by the Examiner.

The Examiner objects to the drawings as failing to show steps. This is highly irregular. To the undersigned's knowledge, the "features" discussed by 37 CFR 1.83(a) refer to the structural features, not steps. Applicant requests clarification and asks that the Examiner verify that this is a proper objection, although additional drawings can be provided if the Examiner insists.

Figures 3-5 have been amended to add the labels as requested by the Examiner.

Claims 1-8 stand rejected as obvious over U.S. Patent No. 5,689,564 ("Divivier") in light of U.S. Patent No. 4,811,284 ("Adler").

Applicant thanks the Examiner for the detailed reasoning of the Office Action, but respectfully traverses the Examiner's argument.

Divivier teaches a pipelined processor with a two-tier prefetch buffer. Valid instruction bytes from the second tier are loaded into the first tier, and instruction bytes from memory for which there is no room in the first tier are loaded into the second tier.

Adler teaches a communication buffer that is part of an input/output (I/O) device. The communication buffer can store a table which lists the size and location of the various messages being processed by the I/O device.

Claim 1 calls for using the validity value and the size value to control an aligner that is coupled to the prefetch buffer and the memory unit to align the requested instructions. Neither Divivier nor Adler teach even the concept of aligning instructions, much less that the validity value or size value could be used to control an aligner.

Claim 1 also calls for storing a size value in said prefetch buffer indicating the number of unissued words remaining in the prefetch buffer, and adjusting the size value in the prefetch buffer depending on the number and size of instructions that are issued in the instruction cycle. The Examiner acknowledges that Divivier does not teach a size value, but suggests that Adler teaches these steps. Applicant respectfully disagrees.

First, Adler teaches only that the controller maintains a table that indicates the number of bytes in each particular message. Adler does not teach that the size value be adjusted on an instruction cycle basis depending on the number and size of the instructions issued that instruction cycle. Rather, Adler teaches taking action only after each message is entirely completed (see column 6, lines 56-68). Thus, the size entries in Adler's table are static. At best, Adler implies that the entire entry is deleted when the message transaction is complete.

Second, the Examiner states that it would have been obvious to use Adler's size values to keep the prefetch buffer as full as possible. However, this makes no sense. Divivier's buffers are filled as soon as any byte is indicated as invalid. How would storing the size value aid in keeping the prefetch buffer full, when the buffer is already kept full by the local validity data?

Third, even if Adler were combined with Divivier, the result would not be the claimed invention. Adler does not teach or suggest anything about using the size value to align the instructions.

Claim 1 also calls for issuing a requested number of instructions, storing a validity value to indicate that the prefetch buffer contains invalid data if all instructions from the prefetch buffer have been issued, and storing a plurality of instructions from the instruction stream in the prefetch buffer in case the prefetch buffer contains invalid data. Divivier does not operate in this fashion. In Divivier, there is a validity bit for each byte in each prefetch buffer. As the Examiner notes, Divivier teaches keeping the prefetch buffer as full as possible. Specifically, if any bytes are marked as invalid, then the instructions for those particular bytes are loaded into the prefetch buffers. In contrast, the method of claim 1, the validity value for the prefetch buffer indicates invalid data if all instructions from the prefetch buffer have been issued. Since Adler does not teach anything about validity values, the combination of Divivier and Adler would not render claim 1 obvious.

For any of the above reason, Applicant submits that the combination of Divivier with Adler cannot render claim 1, and the claims depending therefrom, obvious.

Although Grochowski teaches a multiplexer system which can align instructions from two buffers, Grochowski does not appear to teach storing a size value in the prefetch buffer that indicates the number of unissued words remaining in the prefetch buffer, and does not teach using this size value to set the aligner.

Claim 3 includes limitations similar to those in claim 1, and should be allowable for similar reasons.

Attached is a marked-up version of the changes being made by the current amendment.

Applicant asks that all claims be allowed. Enclosed is a $920.00 check for the Petition for Extension of Time fee.

Please apply any other charges or credits to Deposit Account No. 06-1050.

Respectfully submitted,

Date:_____

_____

David J. Goren
Reg. No. 34,609

Telephone: ~~(650) 322-5070~~
Facsimile: ~~(650) 854-0875~~

ph 650-839-5070
Fax 650-839-5071

Infineon Technologies North America Corp.
c/o Siemens Corporation
Intellectual Property Department
186 Wood Avenue South
Iselin, NJ 08830

Version with markings to show changes made

In the Title:

Please replace the Title with the following:

--APPARATUS AND METHOD FOR ALIGNING VARIABLE-WIDTH INSTRUCTIONS WITH A PREFETCH BUFFER--.

In the claims:

Claims 8-14 have been cancelled.

Claims 1-5 and 7-8 have been amended as follows:

1. (Amended) Method for providing a plurality of aligned instructions from an instruction stream having variable-width instructions provided by a memory unit for execution within a pipelined microprocessor comprising a prefetch buffer, [whereby said prefetch buffer stores prefetched instructions and additional information about the validity and size of said prefetch buffer,] said method comprising the steps of:

for each instruction cycle in which the microprocessor issues one or more instructions,

-in case said prefetch buffer [containing] contains invalid data:

a) requesting an instruction stream from the memory unit and storing a plurality of instructions from said instruction stream in said prefetch buffer;

b) [setting said data for validity in said prefetch buffer] storing a size value in said prefetch buffer indicating the number of unissued words remaining in the prefetch buffer;

c) issuing a requested number of instructions from said [requested instruction stream] memory unit and using the validity value to control an aligner that is coupled to the prefetch buffer and the memory unit to align the requested instructions;

d) depending on [how many] the number and size of instructions that are issued in the instruction cycle, [reducing] adjusting the size [data] value in said prefetch buffer; [, respectively;]

e1) storing a validity value in said prefetch buffer indicating that said prefetch buffer contains valid data if not all instructions from said prefetch buffer have been issued;

e2) [invalidating said] storing a validity value in said prefetch buffer indicating that the prefetch buffer contains invalid data if all instructions from said prefetch buffer have been issued;

-in case said prefetch buffer contains valid data:

f) issuing a requested number of instructions from said prefetch buffer and using the validity value and size value to control the aligner to align the requested instructions;

g) depending on [how many] the number and size of instructions that are issued in the instruction cycle, [reducing] adjusting the [data] size value [for the number of instructions stored] in said prefetch buffer, respectively;

h) [invalidating] changing said validity value to indicate that the prefetch buffer contains invalid data if all instructions from said prefetch buffer have been issued. [;]

2. (Amended) Method according to claim 1, wherein if in step f) the number [of stored] instructions stored in the prefetch buffer is less than the number of requested instructions, [requesting a further instruction stream from said memory unit and] the issuing step includes combining [the necessary] instructions from said [further instruction stream] memory unit with [the] prefetched instructions from said prefetch buffer.

3. (Amended) Method for providing a plurality of aligned instructions from an instruction stream provided by a memory unit for execution within a pipelined microprocessor comprising a first and second prefetch buffer, [whereby said prefetch buffers store prefetched instructions and additional information about the validity and size of said prefetch buffers,] said method comprising the steps of:

storing first and second validity values for said first and second prefetch buffers, respectively;

storing a first size value and a second size value in said first and second prefetch buffers, indicating the number of unissued words remaining in the first and second prefetch buffers, respectively;

for each instruction cycle in which the microprocessor issues one or more instructions,

-in case both of said prefetch buffers contain invalid data:

a) requesting an instruction stream from the memory unit and storing a plurality of instructions from said instruction stream in said first prefetch buffer;

b) [setting said data for validity in said first prefetch buffer c)] issuing a requested number of instructions from said [requested instruction stream] memory unit and using the first and second validity values to control an aligner that is coupled to the first prefetch buffer, the second prefetch buffer and the memory unit to align the requested instructions;

[d] c) depending on [how many] the number and size of instructions that are issued in the instruction cycle, [reducing] adjusting the first size [data] value in said first prefetch buffer,

d) setting the first validity value to indicate that said first prefetch buffer contains valid data if not all instructions from said first prefetch buffer have been issued;

e) [invalidating said] setting the first validity value to indicate that the first prefetch buffer contains invalid data if all instructions from said first prefetch buffer have been issued;

-in case at least one [or both of] said prefetch buffers contains valid data:

f) issuing a requested number of instructions from said at least one prefetch buffer and using the first and second validity values and the first and second size values to control the aligner to align the requested instructions;

g) depending on [how many] the number and size of instructions that are issued in the instruction cycle, [reducing] adjusting at least one of the [data] size values for said at least one [for the number of instructions stored in said] prefetch buffer;

h) [invalidating] setting at least one of said validity values to indicate that said at least one prefetch buffer contains invalid data if all instructions from said at least one prefetch buffer have been issued.

4. (Amended) Method according to claim 3, wherein if in step f) the number of [stored] instructions stored in the first prefetch buffer is less than the number of requested instructions, the issuing step includes combining [the necessary] instructions from [said other] the second prefetch buffer with [the prefetched] instructions from [said one] the first prefetch buffer.

5. (Amended) Method according to claim 3, wherein before step f) the following steps are inserted:

e1) requesting an instruction stream from the memory unit and storing a plurality of instructions from said instruction stream in [the respective] another prefetch buffer that contains invalid data;

e2) setting said [data for] validity value for [in] said another prefetch buffer to indicate that said another prefetch buffer contains valid data.

7. (Amended) Method according to claim 3, comprising after step f), in case of one prefetch buffer containing invalid data, the step of requesting an instruction stream from the memory unit and storing a plurality of instructions from said instruction stream in the respective other prefetch buffer.

8. (Amended) Method according to claim 3 comprising after step c) the step of requesting an instruction stream and storing a plurality of instructions from said instruction stream in the [respective other] second prefetch buffer.

Please add the following claims:

14. Method according to claim 3, wherein aligning the requested instructions includes setting a plurality of multiplexers according to the first and second validity values and the first and second size values.

15. Method according to claim 3, wherein indicating the number of unissued words includes storing the number of issued words.

16. Method according to claim 1, wherein indicating the number of unissued words includes storing the number of issued words.

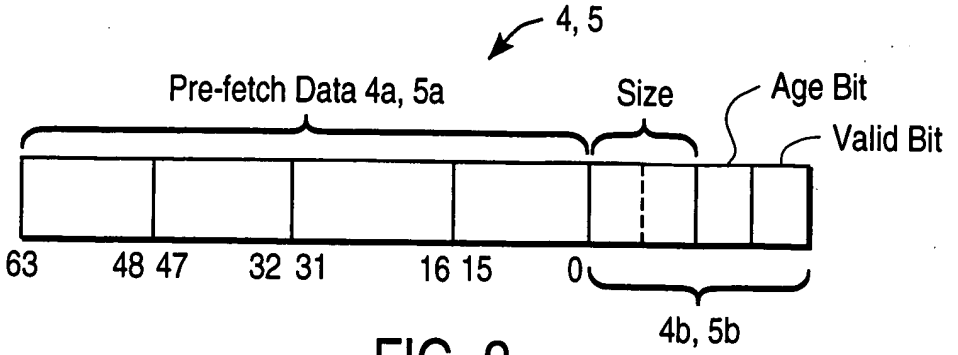

FIG. 2

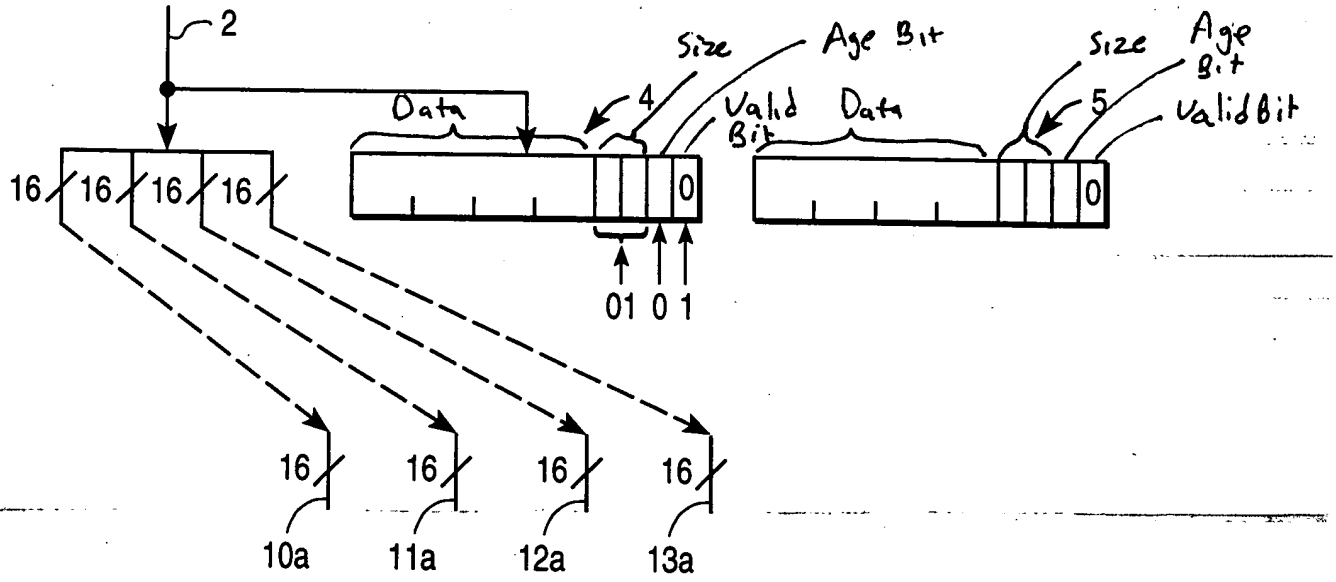

FIG. 3

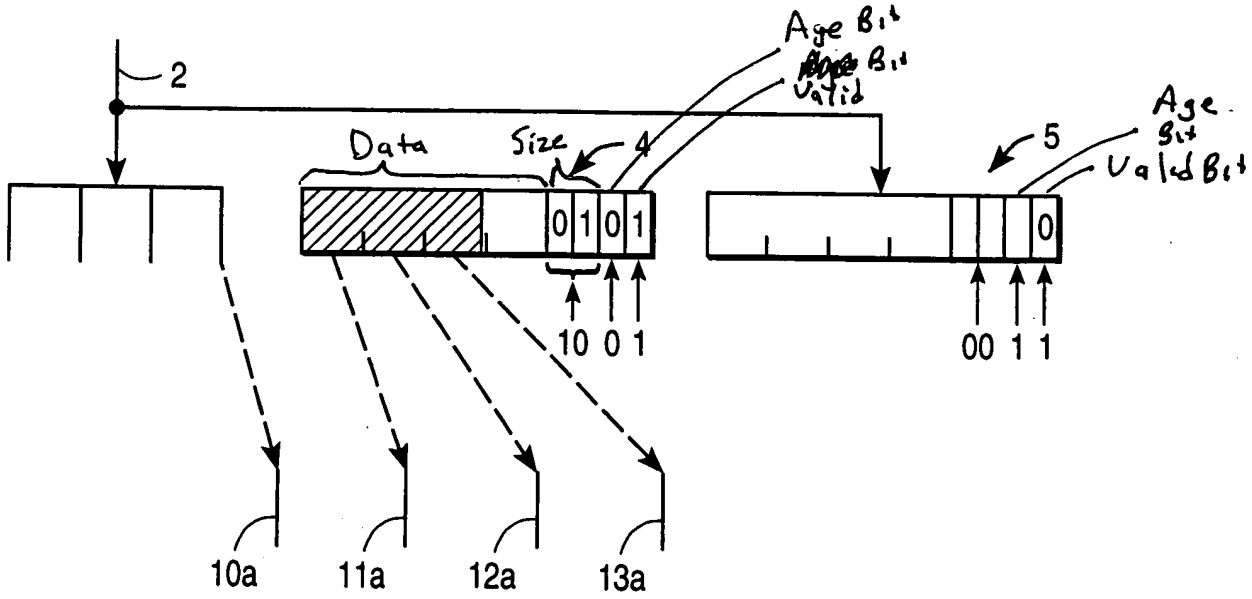

FIG. 4

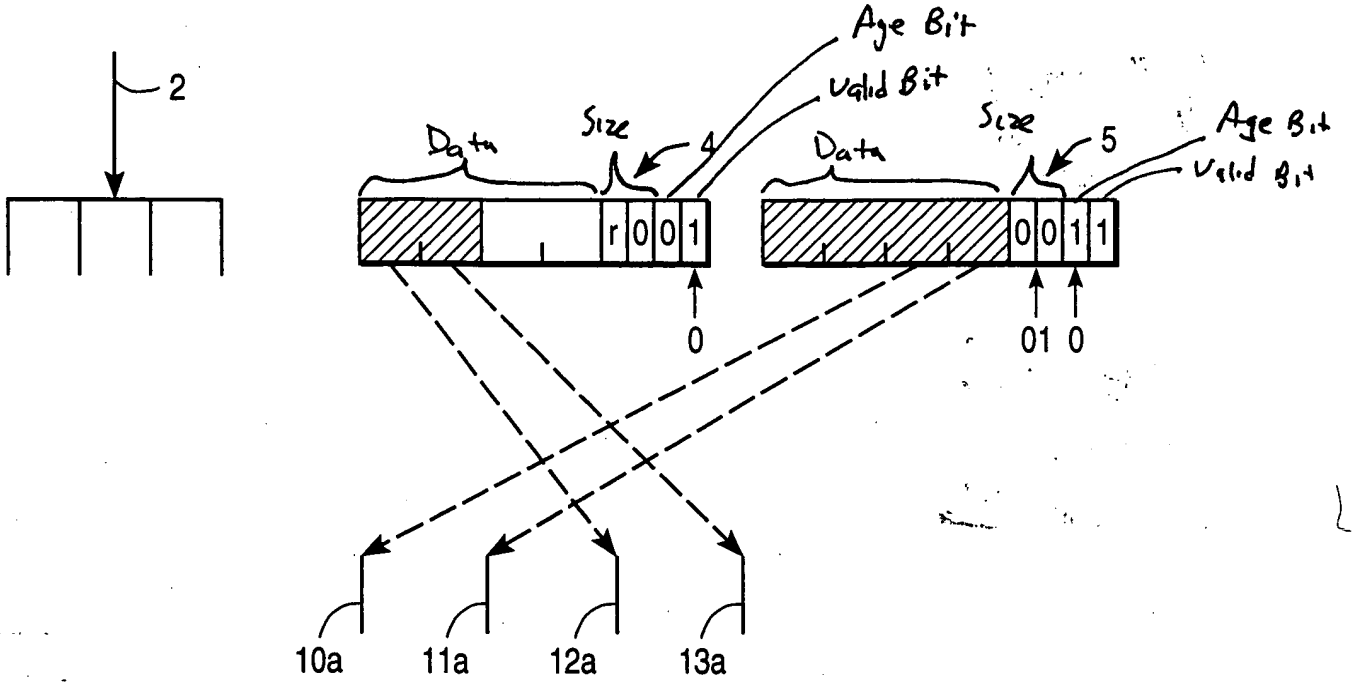
2
Age Bit
Valid Bit
Data
Size
4
r 0 0 1
0
Data
Size
5
Age Bit
Valid Bit
0 0 1 1
01 0
10a
11a
12a
13a

FIG. 5

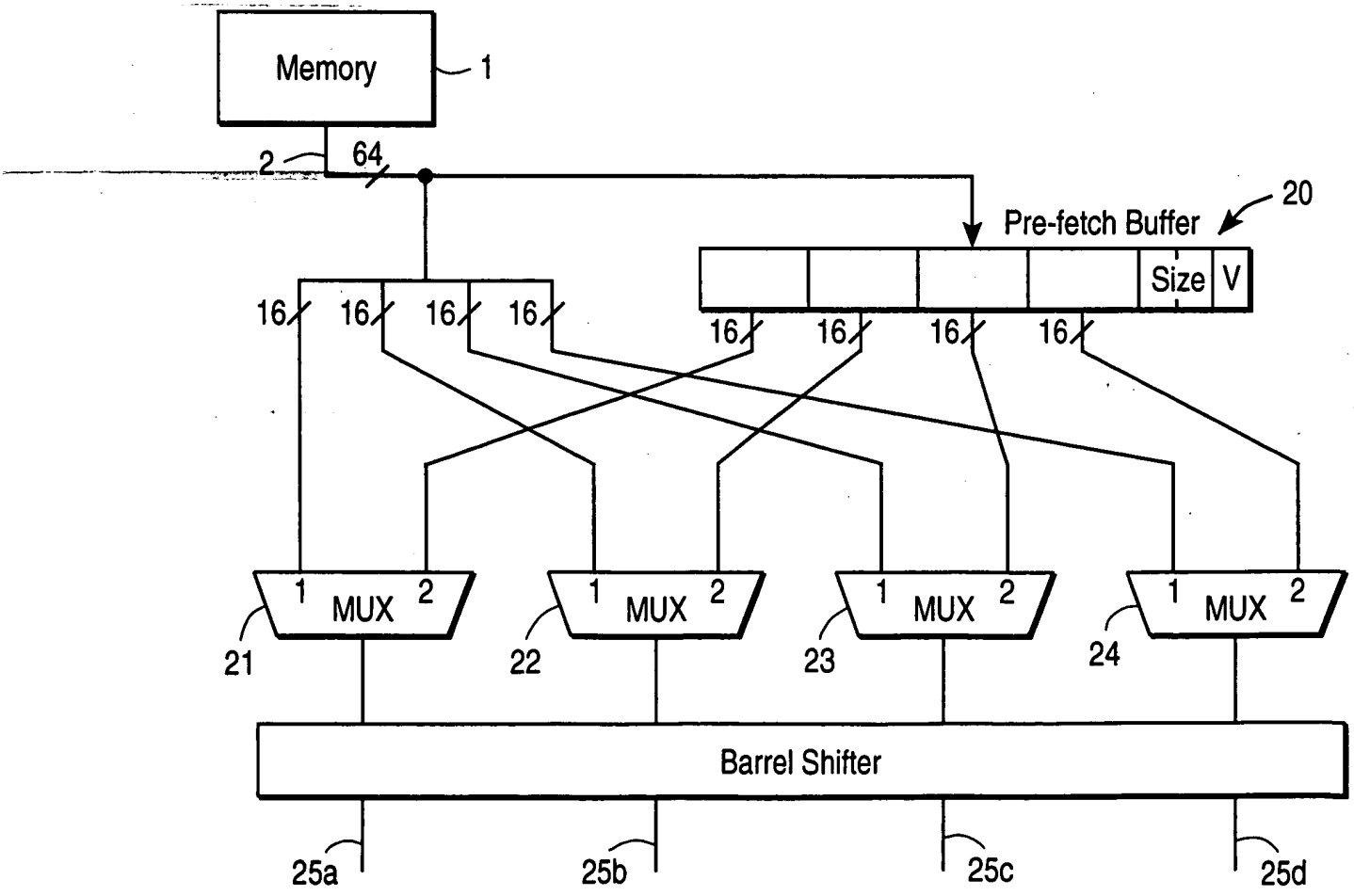
Memory
1
2
64
Pre-fetch Buffer
20
Size
V
16
16
16
16
16
16
16
16
1
MUX
2
21
22
23
24
Barrel Shifter
25a
25b
25c
25d

FIG. 7